



[Advanced Search](#) [Preferences](#) [Language Tools](#) [Search Tips](#)

Tirumalai and pipelining and loop and registers "Code Generation Schema for Modulo S...

The "AND" operator is unnecessary -- we include all search terms by default. [\[details\]](#)
 "Loops" (and any subsequent words) was ignored because we limit queries to 10 words.

[Web](#) - [Images](#) - [Groups](#) - [Directory](#) - [News](#)

Searched the web for Tirumalai and pipelining and loop and registers "Code Generation Schema for Modulo Scheduled Loops - ...

Citations: Code Generation Schema for Modulo Scheduled Loops - ...

... of the kernel loop is unnecessary because ... uses a software pipelining algorithm called ... and

PP Tirumalai, "Code Generation Schema for Modulo-Scheduled Loops," in ...

citeseer.nj.nec.com/context/16222/593666 - 14k - [Cached](#) - [Similar pages](#)

Lifetime-sensitive Modulo Scheduling in a Production Environment ...

... if a schedule requires more registers than those ... 1996 13 A mathematical formulation

of the loop pipelining problem - Cortadella ... (context) - Tirumalai, Lee et ...

citeseer.nj.nec.com/lossa01lifetimesensitive.html - 28k - [Cached](#) - [Similar pages](#)

[[More results from citeseer.nj.nec.com](#)]

Register allocation for software pipelined loops

... YT Hsu , Joseph P. Bratt, Overlapped loop support in ... 11 M. Lam, Software pipelining: an effective scheduling technique ... 19 P. Tirumalai , M. Lee , M. Schlansker ...

[portal.acm.org/citation.cfm?](http://portal.acm.org/citation.cfm?id=143141&jmp=indexterms&dl=portal&dl=ACM&CFID=11111111&C...)

[id=143141&jmp=indexterms&dl=portal&dl=ACM&CFID=11111111&C...](http://portal.acm.org/citation.cfm?id=143141&jmp=indexterms&dl=portal&dl=ACM&CFID=11111111&C...) - [Similar pages](#)

[PS] Software Pipelining

File Format: Adobe PostScript - [View as Text](#)

... By Rau/Schlansker/Tirumalai. ... Software Pipelining Allan, Jones, Lee, Allan. ... So, the loop kernel is composed of one instruction, which executes all the following ...

www.wisdom.weizmann.ac.il/~verify/events/2001/weekly/lecture1.ps - [Similar pages](#)

[PPT] Overview

File Format: Microsoft Powerpoint 97 - [View as HTML](#)

... Loop optimizations like software pipelining could be applied ... code generated shows no evidence of software pipelining. ... PP.Tirumalai B. Ramakrishna Rau, Michael S ...

www.cse.iitd.ernet.in/esproject/homepage/docs/projects/2002-2003/diviya.final.ppt - [Similar pages](#)

Software pipelining: an effective scheduling technique for VLIW ...

... references. 1 Aiken, A. and Nicolau, A. Perfect Pipelining: A New Loop Parallelization Technique. Cornell University, Oct., 1987. ...

www.acm.org/pubs/articles/proceedings/pdidi/53990/p318-lam/p318-lam.pdf - 101k - [Cached](#) - [Similar pages](#)

[PDF] CRED: Code Size Reduction Technique and Implementation for ...

File Format: PDF/Adobe Acrobat - [View as HTML](#)

... operation corresponds to a software pipelining operation. ... conditionally executing the schedule of loop body ... we can decide the conditional registers required for ...

www.utdallas.edu/~qfzhuge/papers/escodes02-workshop.pdf - [Similar pages](#)

[PDF] Optimal Code Size Reduction for Software-Pipelined and Unfolded ...

File Format: PDF/Adobe Acrobat - [View as HTML](#)

... we need to use four conditional registers to completely ... to be inserted into the loop body for ... be parallelized with other instructions by software pipelining. ...

www.utdallas.edu/~qfzhuge/papers/iss02.pdf - [Similar pages](#)

[[More results from www.utdallas.edu](#)]

Sponsored Links

[IronWorks Code Generator](#)

Generate n-tier .NET apps with 51 customizable templates. Free trial.
www.eSolutionsPR.com
 Interest: [*****](#)

[Metal Registers & Grilles](#)

Vintage style brass & cast iron registers, grilles, & vent covers.
www.shop4classics.com
 Interest: [*****](#)

[.NET Code Generation Tool](#)

Save time on any C++, C# and VB .NET project. Free trial!
www.workstate.com
 Interest: [*****](#)

[RAD Code Generation tool](#)

Template based code generation ASP, Perl, VB, C# over 50 templates
www.lockwoodtech.com
 Interest: [*****](#)

[Code Generation in Action](#)

Learn how to build code generators for a range of applications domains
www.manning.com/herrington
 Interest: [*****](#)

[Tirupati Venkateswara VCD](#)

Video of the famous Balaji temple in Tirupati during Brahmotsavam
www.matchless-gifts.com
 Interest: [*****](#)

[Code Generation Tools](#)

Arbitrary spec or legacy languages Many target languages, optimization
www.semanticdesigns.com
 Interest: [*****](#)

[Generate Source Code](#)

Workstart from Westfaro Corporation simplifies source code generation.
www.westfaro.com
 Interest: [*****](#)

[See your message here...](#)

[PDF] [A Register File and Scheduling Model for Application Specific ...](#)

File Format: PDF/Adobe Acrobat - [View as HTML](#)

... simple logic units can be used to activate proper **registers** at run ... schedules cyclic DFGs (Data Flow Graphs) with resource constraints using **loop pipelining**. ...

www.sigda.org/Archives/ProceedingArchives/Dac/Dac96/papers/1996/dac96/pdffiles/04_2.pdf - [Similar pages](#)

[PDF] [Controlling Code Size of Software-Pipelined Loops On the ...](#)

File Format: PDF/Adobe Acrobat - [View as HTML](#)

... support beyond the availability of static predicate **registers**. ... 3.2 Safety Considerations
For a **loop** to be eligible for software **pipelining**, the compiler ...

www.pdcl.eng.wayne.edu/msp01/paper4.pdf - [Similar pages](#)

Google

Result Page: 1 2 3 [Next](#)

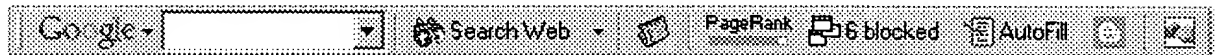
Tirumalai and pipelining and loop an

Google Search

[Search within results](#)

Dissatisfied with your search results? [Help us improve.](#)

Get the [Google Toolbar](#):



[Google Home](#) - [Advertise with Us](#) - [Business Solutions](#) - [Services & Tools](#) - [Jobs, Press, & Help](#)

©2003 Google

120 citations found. Retrieving documents...

B. R. Rau. *Iterative modulo scheduling: An algorithm for software pipelining loops.* In Proc. of the 27th Annual International Symposium on Microarchitecture, pages 63–74, November 1994.

CiteSeer [Home/Search](#) Document Not In Database [Summary](#) [Related Articles](#) [Check](#)

This paper is cited in the following contexts:

[First 50 documents](#) [Next 50](#)

Modulo Scheduling with Integrated Register Spilling - For Clustered Vliw (Correct)

...[1] for the generation of software pipelined schedules. **Modulo scheduling** [26] is a class of software pipelining algorithms that is very cost effective and has been implemented in many production compilers. **Most of the early modulo scheduling techniques focused mainly on achieving high throughput** [1, 7, 25, 28]. However, one of the drawbacks of modulo scheduling (and software pipelining in general) is that they increase the register requirements. **This** has motivated some recent modulo scheduling approaches that not only try to maximize throughput but also try to minimize register requirements [6, 9, 16, ...

...forcing a node in a particular cycle, the heuristic ejects nodes that cause resource conflicts with the forced node. **If** for a particular resource conflict several candidate nodes are possible, the heuristic selects the one that was first placed in the partial schedule *S*. **Other iterative algorithms** [6, 16, 28] **eject all the operations that cause a resource conflict**. In our iterative algorithm, only one is ejected. **The** heuristic also ejects all previously scheduled predecessors and successors whose dependence constraints are violated due to the enforced placement. **Notice** that all the unscheduled

B. R. Rau. Iterative modulo scheduling: An algorithm for software pipelining loops. In Proc. of the 27th Annual International Symposium on Microarchitecture, pages 63–74, November 1994.

Exploiting Pseudo-schedules to Guide Data.. - Aleta. Codina.. (Correct)

...of the register pressure, the pressure on the register buses, and the resource constraints for each cluster. **MI** is the maximum between (the initiation interval due to resources) and (the initiation interval due to recurrences) these two values are computed as in [32]. Then, instructions are scheduled according to their computed cluster assignment. **If** an instruction cannot be scheduled in the assigned cluster, the instruction is moved to a different cluster. **If** an instruction cannot be scheduled in any cluster, the *II* is increased, the partition is modified

..., where (6720 63804 0 157 158 159 7 : and where *C E8F,1HG* is the number of communications necessary to schedule the partition, *CKJ LMGON G* is the number of buses in the architecture and *J LMGUTW3 V* is the latency of the buses. **To compute ; we proceed as in [32], but also take into account the latency of the edges between instructions in different clusters.** Then, assuming *X Y 8 Z = #* , we try to find a suitable slot for each node. **Since** the pseudo schedule needs to be computed as accurately as possible, nodes are scheduled using the same

[Article contains additional citation context not shown here]

B. Rau. Iterative Modulo Scheduling: An Algorithm for Software Pipelining Loops. In Proc. of 27th Int. Symp. on Microarchitecture, pages 67–74, Nov 1994.

MIRS : Modulo Scheduling with Integrated Register Spilling - Zalamea, Llosa, Ayguadé.. (Correct)

...scheduling technique able to exploit this ILP out of a loop by overlapping operations from various successive loop iterations. **Different** approaches have been proposed in the literature [2] for the generation of software pipelined schedules. **Some of them mainly focus on achieving high throughput** [1, 13, 18, 25, 26, 28]. This work has been supported by the Ministry of Education of Spain under contract TIC 98 511, and by CEPBA (European Center for Parallelism of Barcelona) Javier Zalamea is granted by the Agencia Espa nola de Cooperaci on Internacional. **Register** allocation consists in finding the final

...paper presents a novel approach for register spilling in modulo scheduled loops. **In** this approach, instruction scheduling, register allocation, and register spilling are simultaneously in the same step. **To achieve this, it uses the ability of some previous iterative modulo scheduling techniques** [12, 13, 17, 28] **to backtrack, i.e. to undo previous scheduling decisions and reschedule operations.** In order to have reasonable low spill code requirements, MIRS is based on HRMS (Hypernode Reduction Modulo Scheduling [23] a register sensitive modulo scheduler. **Our** proposal is compared with the ideal case

[Article contains additional citation context not shown here]

B. R. Rau. Iterative modulo scheduling: An algorithm for software pipelining loops. In Proc. of the 27th Annual International

An Overview of the Intel IA-64 Compiler - Carole Dulong Microcomputer (Correct)

... of values [4, 6] In IA 64, on the other hand, unrolling of the kernel loop is unnecessary because rotating registers can be used to perform renaming of the registers, thus reducing the code size [5, 6, 7] The Intel IA 64 compiler uses a software pipelining algorithm called modulo scheduling [8]. In modulo scheduling, a minimum candidate II is computed prior to scheduling. This candidate II is the maximum of the resource constrained minimum II and the recurrenceconstrained (dependence cycle constrained) minimum II. **prolog epilog kernel loop** Figure 25: Execution phases in

B. R. Rau, "Iterative Modulo Scheduling: An Algorithm for Software Pipelining Loops," in Proceedings of the 27th International Symposium on Microarchitecture, December 1994, pp. 63-74.

Genetic Programming Applied to Compiler Heuristic.. - Stephenson, O'Reilly, ... (Correct)

...functions to achieve suitable performance [2] Priority functions are widely used and tied to complicated factors. **A non exhaustive list of examples, just in compilation, includes list scheduling [9] clustered scheduling [14] hyperblock formation [12] meld scheduling [1] modulo scheduling [17] and register allocation [6] GP s representation appears ideal for improving priority functions.** We have tested this observation via two case studies: predication and register allocation. **Predication** Studies show that branch instructions account for nearly 20 of all instructions executed in a

B. R. Rau, *Iterative Modulo Scheduling: An Algorithm for Software Pipelining Loops*. In Proceedings of the 27th Annual International Symposium on Microarchitecture (MICRO-2J), November 1994.

Meld Scheduling: A Technique for Relaxing Scheduling.. - Abraham, Kathail, Deitrich (1998) (Correct)

...provide greater improvement on floating point benchmarks as compared to integer benchmarks. **However**, floating point benchmarks are highly loop intensive and inter region dangles are less of a problem, since most of the performancecritical dangles occur at the back edges. **Modulo scheduling of loops [10 13] is capable of handling these dangles during scheduling.** Loop unrolling provides similar benefits. In our experiments, loops were unrolled eight times reducing the impact of inter region dangles on the overall performance. **For example**, the average size of the superblocks was 88 operations for

B. R. Rau, "Iterative modulo scheduling: an algorithm for software pipelining loops," presented at 27th Annual International Symposium on Microarchitecture, San Jose, California, 1994.

Genetic Programming Applied to Compiler Heuristic.. - Stephenson, O'Reilly, ... (Correct)

...functions to achieve suitable performance [2] Priority functions are widely used and tied to complicated factors. **A non exhaustive list of examples, just in compilation, includes list scheduling [9] clustered scheduling [14] hyperblock formation [12] meld scheduling [1] modulo scheduling [17] and register allocation [6] GP s representation appears ideal for improving priority functions.** We have tested this observation via two case studies: predication and register allocation. **4** Predication Studies show that branch instructions account for nearly 20 of all instructions executed in

B. R. Rau, *Iterative Modulo Scheduling: An Algorithm for Software Pipelining Loops*. In Proceedings of the 27th Annual International Symposium on Microarchitecture (MICRO-24), November 1994.

Region-based Register Allocation for EPIC Architectures - Kim (Correct)

...scheduling, the data dependency graph is useful for guiding optimizations that consider critical path lengths in a program. 2.3.3.2.4 Modulo scheduling and Rotating register allocations The loop scheduling consists of two modules: the modulo scheduler and stage scheduler. **The modulo scheduler [41] [50] allocates resources for the loop kernel subject to an initiation interval.** The stage scheduler moves operations across stages in order to reduce register usage in the loop. **When a loop is modulo scheduled**, some of the virtual registers in the loop are designated as rotating registers.

...on live range splitting (FBS FBR) The percentage of savings are showed in the last column. 4.5 Live range split for predicated codes Predication[27] has been included in EPIC style architectures and provides many opportunities of ILP optimization to the compiler. **It enables modulo scheduling[41] to reduce code expansion and to be scheduled with kernel only codes.** More corn 91 Benchmark BASE FBS FBS FBR 1 FBS BASE 1 FBS FBR FBS 008.espresso 1487 744 717 49.97 3.63 023.eqntott 359 251 239 30.08 4.78 072.sc 352 115 115 67.33 0.00 085.gcc 7431 2312 2078 68.89 10.12

B. Rau, *Iterative modulo scheduling: An algorithm for software pipelining loops*. Proceedings of the 27th Annual Symposium on Microarchitecture, December 1994.

Meta Optimization: Improving Compiler Heuristics... - Stephenson... (2002) (Correct)

...this paper we use the formation of predicated hyperblocks as a case study. **Meld scheduling:** Abraham et. al rely on a priority function to schedule across region boundaries [1] The priority function is used to sort regions by the order in which they should be visited. **Modulo scheduling:** In [19], Rau states, **As is the case for acyclic list scheduling, there is a limitless number of priority functions that can be devised for modulo scheduling.** Rau describes the tradeoffs involved when considering scheduling priorities. **Register allocation:** Many register allocation algorithms use cost

B. R. Rau. *Iterative Modulo Scheduling: An Algorithm for Software Pipelining Loops*. In Proceedings of the 27th Annual International Symposium on Microarchitecture (MICRO-24), November 1994.

Exploiting Schedule Slacks for Rate-Optimal... - Yang, Govindarajan, ... (2002) (Correct)

...cycles. **The objective of a software pipelining method is to construct a schedule that has a high computation rate, or equivalently a low II. In the past, resource constrained software pipelining has been studied extensively by several researchers and a number of modulo scheduling algorithms [7, 13, 16, 22] have been proposed.** For a comprehensive survey of software pipelining methods the reader is referred to [21] This paper presents a new power aware software pipelining method for VLIW architectures, which can minimize the power consumption of software pipelined loops without sacrificing

...schedule using integer linear programming formulation. **Our work does not require such hardware support for frequency and voltage scaling. A relevant work in power aware software pipelining is by Yun et al. [28] This work introduces certain modifications to the iterative modulo scheduling algorithm [22] to minimize step power the difference in the power consumed between two consecutive time steps for a software pipelined loop.** Their objective is to derive a schedule under which power consumption is better balanced under a VLIW architecture. **This is different from our objective which**

B. Ramakrishna Rau. *Iterative modulo scheduling: An algorithm for software pipelining loops*. In Proc. of MICRO-27, pages 63–74, San Jose, Calif., Nov.–Dec. 1994.

Power-Performance Trade-Offs For Energy-Efficient... - Yang... (2002) (Correct)

...to re order a schedule made by some performance oriented scheduling algorithm to achieve energy saving with minimal performance degradation. **A relevant work by Yun et al. [20] targets poweraware software pipelining. They proposed a heuristic algorithm which extended iterative modulo scheduling [13], tries to minimize step power for a software pipelined loop on a cycle by cycle basis.** 6. **Conclusions** trade offs in the design space of energy efficient architectures. **It studies the interplay between lowpower architecture features and compiler optimization techniques, specifically software**

B. R. Rau. *Iterative modulo scheduling: An algorithm for software pipelining loops*. In Proc. of MICRO-27, pages 63–74, San Jose, Calif., Nov.–Dec. 1994.

Global Software Pipelining with Iteration Preselection - Gregg (2000) (Correct)

...a time are similar to early attempts at global acyclic scheduling which percolated operations from one basic block to the next without any final intended destination. **Moving an operation one iteration at a time may create a temporarily worse schedule, that can later be transformed into a better one [Rau94].** The problem is to distinguish between such good moves, and shifts that genuinely make the schedule worse. **Despite the problems of loop shifting, it has one very important advantage it extends naturally to software pipelining loops containing branches. Loop shifting algorithms can pipeline**

B. Ramakrishna Rau. *Iterative modulo scheduling: An algorithm for software pipelining loops*. In 27th Annual International Conference on Microarchitecture. ACM, December 1994.

Optimal Loop Scheduling for Hiding Memory Latency Based ... - Wang, O'Neil, Sha... (2000) (Correct)

...the loop. It can result in high performance code but increased register requirements [10] Rau and Eichenberger have done research on optimum modulo schedules, taking into consideration the minimum register requirement. **They consider not only the data flow, but also the control flow of the program [7, 18].** None of the above research efforts, however, includes the prefetching idea or considers the data fetching latency in their algorithms. **We will restrict our study to nested loops with uniform data dependencies. Even if most loop nests have data dependencies, the study of uniform loop nests is**

B. R. Rau. *Iterative modulo scheduling: An algorithm for software pipelining loops*. In Proceedings of the 27th Annual International Symposium on Microarchitecture, pages 63–74, Nov., 1994.

Maximizing Pipelined Functional Units Usage for Minimum... - Yang, Govindarajan, Cai (2001) (Correct)

...pipelining is an important compilation technique applied on loops to exploit instruction level parallelism. **In the past, resource constrained software pipelining has been studied extensively by several researchers and a number of modulo scheduling algorithms have been proposed in the literature [6, 16, 21, 31].** The objective of a software pipelining method is to construct a schedule that satisfies both the resource constraints of the architecture and the dependence constraints imposed by the program, such that the constructed schedule has a very low initiation interval (II) The schedule which achieves

...available resources. **This applies not only to critical instructions, those that are on critical recurrence cycle(s) or those that use critical resource(s) but also to all other instructions as well. In certain software pipelining methods, instructions are scheduled at the earliest possible time [31].** However, issuing instruction as early as possible may schedule non critical instructions along with critical instructions at the same time step, requiring multiple instances of functional units to be active simultaneously. **As explained earlier, since we assume a power model in which all or none**

[Article contains additional citation context not shown here]

B. Ramakrishna Rau. *Iterative modulo scheduling: An algorithm for software pipelining loops*. In Proc. of the 27th Ann. Intl. Symp. on Microarchitecture, pages 63--74, San Jose, Calif., Nov. 30--Dec.2, 1994.

Minimizing Average Schedule Length under Memory Constraints.. - Wang, O'Neil, Sha (2000) (Correct)

...the loop. It can result in high performance code but increased register requirements [10] Rau and Eichenberger have done research on optimum modulo schedules, taking into consideration the minimum register requirement. **They consider not only the data ow, but also the control ow of the program [8,18].** None of the above research e orts, however, includes the prefetching idea or considers the data fetching latency in their algorithms. **DO 10 n1 = 1 , N1 DO 20 n2 = 1, N2 y (n1 , n2) x (n1 , n2) c (0 , 1) y (n1 , n2 1) c (0 , 2) y (n1 , n2 2) c (1 , 0) y (n1 1**

B. R. Rau. *Iterative modulo scheduling: An algorithm for software pipelining loops*. In Proceedings of the 27th Annual International Symposium on Microarchitecture, pages 63-74, Nov, 1994. 30

Enhancing Loop Buffering of Media and Telecommunications.. - Sias, Hunter, Hwu (2001) (1 citation) (Correct)

... and StarCore 140) are based on a VLIW design paradigm, with good reason: **VLIW offers wide issue (today up to eight operations per cycle) with relatively little instruction issue overhead, clustering is natural and offers enhanced scalability [1] and compiler techniques such as software pipelining [2] effectively employ the VLIW s many processing units in a wide variety of loop kernels.** In the embedded market, where power margins dictate use of the lowest possible clock frequency to achieve a given processing rate, cycles cannot be wasted waiting for branch resolution and instruction fetch.

B. R. Rau, "Iterative modulo scheduling: An algorithm for software pipelining loops," in Proc. 27th Int'l Symposium on Microarchitecture, pp. 63--74, Dec. 1994.

Power-Aware Modulo Scheduling for High-Performance VLIW - Yun (Correct)

...SCHEDULING OVERVIEW Software pipelining is an aggressive loop scheduling technique for VLIW processors. **It transforms a sequential loop so that new iterations can start before preceding ones finish, thus overlapping the execution of multiple iterations in a pipelined fashion. Modulo scheduling [6, 13] is one of the scheduling algorithms for implementing software pipelining.** 1 Since a large number of loops contain no conditionals, we concentrate on loops with no control flows in this paper. **For loops with control flows, we assume a hardware mechanism that supports predicated execution.**

...high level synthesis [4] and logic level synthesis [8] 2 Such a graph is called a data flow graph (DFG) in the context of synchronous VLSI circuits. 3 For simplicity, we assume that the functional units are fully pipelined. **Complex resource constraints can be handled by resource reservation table [13].** 2) $r2 = op2(r1, r5)$ 1) $r1 = op1(r3)$ 3) $r3 = op3(r2)$ 4) $r4 = op4(r3)$ 5) $r5 = op5(r2)$ 6) $r6 = op6(r6)$ a) NOP NOP NOP NOP NOP NOP (1) 2) 3) n n n (6) n (4) n 1 (5) n 1 NOP NOP NOP NOP (1) 2) 3) n n n (6) n (5) n (4) n 1 NOP NOP (b) 1) 2) 3) 0 0 1 0 0 1 1 (6) c) d)

[Article contains additional citation context not shown here]

B. R. Rau. *Iterative Modulo Scheduling: An Algorithm for Software Pipelining Loops*. In Proc. of the 27th Annual International Symposium on Microarchitecture (Micro-27), pages 63--74, 1994.

Adapting Software Pipelining for Reconfigurable Computing - Callahan, Wawrzynek (2000) (2 citations) (Correct)

...4.1 Modulo Scheduling Because multiple iterations will be active simultaneously on the network of modules, care must be taken so that nonqueue memory accesses from different iterations do not conflict. **We utilize a scheduling algorithm directly based on Rau s iterative modulo scheduling (IMS) [17].** Modulo scheduling is a framework for scheduling a single iteration of a loop in

a way that resolves resource conflicts among consecutive overlapping iterations [9] Successive iterations have identical schedules, shifted by a number of cycles called the initiation interval (II) Resource

Rau, B. R. *Iterative Modulo Scheduling: An Algorithm for Software Pipelining Loops*. In Proceedings of the 27th Annual International Symposium on Microarchitecture. (1994), ACM, pp. 63--74.

Improved Spill Code Generation for Software Pipelined.. - Zalamea, Llosa.. (1999) (1 citation) (Correct)

....exploits instruction level parallelism (ILP) out of a loop by overlapping operations from various successive loop iterations. **Different** approaches have been proposed in the literature [2] for the generation of software pipelined schedules. **Some of them mainly focus on achieving high throughput** [1, 13, 18, 24, 25, 27]. The main drawback of these aggressive scheduling techniques is their high register requirements [21, 23] Using more registers than available requires some actions which reduce the register pressure but may also degrade the performance (either due to the additional cycles in the schedule or due

....phase. **The II is bounded either by recurrence circuits in the dependence graph of the loop (RecMII) or by resource constraints of the target architecture (ResMII)** The lower bound on the II is termed the Minimum Initiation Interval ($MII = \max(RecMII; ResMII)$) The reader is referred to [13, 27] for an extensive dissertation on how to calculate RecMII and ResMII. In order to perform software pipelining, the Hypernode Reduction Modulo Scheduling (HRMS) heuristic [22] is used. HRMS is a software pipeliner that achieves the MII for a large percentage of the workbench considered in this

B. R. Rau. *Iterative modulo scheduling: An algorithm for software pipelining loops*. In Proc. of the 27th Annual International Symposium on Microarchitecture, pages 63--74, November 1994.

CALiBeR: A Software Pipelining Algorithm for Clustered.. - Akturan, Jacome (2001) (1 citation) (Correct)

....shown in Figure 3(b) The total number of pipe stages (i.e. iterations executing concurrently) on a software pipelined loop body is denoted by P. **The total number of execution steps required by any such (balanced) pipe stage corresponds to the initiation interval (II) of the retimed loop body** [4]. That is, a new iteration is started concluded every II steps. For the example, in Figure 3(b) the initiation interval and the total number of pipe stages are $II=2$ and $P=2$, respectively. **Naturally**, the key objective of software pipelining retiming is to decrease II, thus increasing the execution

B.R. Rau, "Iterative modulo scheduling an algorithm for software pipelining loops", MICRO-27, 1994.

Software and Hardware Techniques to Optimize.. - Zalamea, Llosa.. (2001) (1 citation) (Correct)

....niques have been proposed to efficiently exploit the parallelism available in iterative program constructs. **Software pipelining** [4] [5] is a loop scheduling technique that extracts parallelism from loops by overlapping operations from various consecutive iterations. **Modulo scheduling** [6] [7] is a class of software pipelining algorithms which has been incorporated in many production compilers. In a modulo scheduled loop, the Initiation Interval (II) is the number of cycles between the initiation of successive iterations. For a loop, the lower the II the higher the number of operations

....proven to be very effective [14] In [15] the authors present an approach that improves the performance by simultaneously performing instruction scheduling, register allocation, and register spilling. **To achieve this, it uses the ability of some previous iterative modulo scheduling techniques** [6] [7], [10] [16] to backtrack, i.e. to undo previous scheduling decisions and reschedule operations. Sections II.B and C overview the main proposals in this direction. On the other hand, the organization and management of the register file has been a subject of research in the past. **The main idea**

B. R. Rau, "Iterative modulo scheduling: An algorithm for software pipelining loops," in Proc. of the 27th Annual International Symposium on Microarchitecture, November 1994, pp. 63--74.

Two-level Hierarchical Register File Organization.. - Zalamea, Llosa.. (2000) (4 citations) (Correct)

....exploit the ILP available in programs [4, 12, 20] Loops are the main time consuming part of numerical programs. **Software pipelining** [5, 14] is a loop scheduling technique that extracts parallelism from loops by overlapping operations from various consecutive iterations. **Modulo scheduling** [8, 22] is a class of software pipelining algorithms which has been incorporated in many production compilers. In a modulo scheduled loop, the Initiation Interval (II) is the number of cycles between the initiation of successive iterations. For a loop, the lower the II the higher the number of

B. R. Rau. *Iterative modulo scheduling: An algorithm for software pipelining loops*. In Proc. of the 27th Annual International Symposium on Microarchitecture, pages 63--74, November 1994.

Modulo Scheduling with Integrated Register Spilling.. - Zalamea, Llosa.. (Correct)

....[1] for the generation of software pipelined schedules. **Modulo scheduling** [26] is a class of software pipelining algorithms that is very cost effective and has been implemented in many production compilers. **Most of the early modulo scheduling techniques focused mainly on achieving high throughput** [1, 7, 25, 28]. However, one of the drawbacks of modulo scheduling (and software pipelining in general) is that they increase the register requirements. **This** has motivated some recent modulo scheduling approaches that not only try to maximize throughput but also try to minimize register requirements [6, 9, 16,

....forcing a node in a particular cycle, the heuristic ejects nodes that cause resource conflicts with the forced node. If for a particular resource conflict several candidate nodes are possible, the heuristic selects the one that was first placed in the partial schedule S. **Other iterative algorithms** [6, 16, 28] **eject all the operations that cause a resource conflict**. In our iterative algorithm, only one is ejected. The heuristic also ejects all previously scheduled predecessors and successors whose dependence constraints are violated due to the enforced placement. **Notice** that all the unscheduled

B. R. Rau. *Iterative modulo scheduling: An algorithm for software pipelining loops*. In Proc. of the 27th Annual International Symposium on Microarchitecture, pages 63--74, November 1994.

Lifetime-sensitive Modulo Scheduling in a Production.. - Llosa, Ayguade.. (2001) (1 citation) (Correct)

....require a prohibitive time to construct the schedules and therefore their applicability is restricted to very small loops. **Therefore**, practical algorithms use some heuristics to guide the scheduling process. **Some of the proposals in the literature only care about achieving high throughput** [11,19,20,31,32,37] **while other proposals have also been targeted towards minimizing the register requirements** [9,12,18,24] **which result in more effective schedules**. Stage Scheduling [12] is not a whole modulo scheduler by itself but a set of heuristics targeted to reduce the register requirements of any given

....but lower register requirements. **Unfortunately** there are constraints in the movement of operations that might yield to suboptimal reductions of the register requirements. **Similar heuristics have been 3 included in the IRIS [9] scheduler, which is based on the Iterative Modulo Scheduling** [11,31], **in order to reduce the register pressure at the same time as the scheduling is performed**. Slack Scheduling [18] is a heuristic technique that simultaneously schedules some operations late and other operations early with the aim of reducing the register requirements and achieving maximum

[Article contains additional citation context not shown here]

B.R. Rau. *Iterative modulo scheduling: An algorithm for software pipelining loops*. In Proc. of the 27th Annual Internat. Symp. on Microarchitecture, pages 63-74, November 1994.

Scheduling Time-Constrained Instructions on Pipelined.. - Leung, Palem, Pnueli (Correct)

No context found.

Rau, B. *Iterative modulo scheduling: An algorithm for software pipelining loops*. Proceedings of the 27th Annual Symposium on Microarchitecture (December 1994).

[First 50 documents](#) [Next 50](#)

[Online articles have much greater impact](#) [More about CiteSeer](#) [Add search form to your site](#) [Submit documents](#) [Feedback](#)

CiteSeer - citeseer.org - [Terms of Service](#) - [Privacy Policy](#) - Copyright © 1997-2002 NEC Research Institute



[Advanced Search](#) [Preferences](#) [Language Tools](#) [Search Tips](#)

Tirumalai and pipelining and loop and registers "Code Generation Schema for Modulo S...

The "AND" operator is unnecessary --- we include all search terms by default. [\[details\]](#)
 "Loops" (and any subsequent words) was ignored because we limit queries to 10 words.

[Web](#) - [Images](#) - [Groups](#) - [Directory](#) - [News](#)

Searched the web for Tirumalai and pipelining and loop and registers "Code Generation Schema for Modulo Scheduled Loops"

Citations: Code Generation Schema for Modulo Scheduled Loops - ...

... of the kernel loop is unnecessary because ... uses a software pipelining algorithm called ... and

PP Tirumalai, "Code Generation Schema for Modulo-Scheduled Loops," in ...

citeseer.nj.nec.com/context/16222/593666 - 14k - [Cached](#) - [Similar pages](#)

Lifetime-sensitive Modulo Scheduling in a Production Environment ...

... if a schedule requires more registers than those ... 1996 13 A mathematical formulation

of the loop pipelining problem - Cortadella ... (context) - Tirumalai, Lee et ...

citeseer.nj.nec.com/llosa01lifetimesensitive.html - 28k - [Cached](#) - [Similar pages](#)

[[More results from citeseer.nj.nec.com](#)]

Register allocation for software pipelined loops

... YT Hsu , Joseph P. Bratt, Overlapped loop support in ... 11 M. Lam, Software pipelining: an effective scheduling technique ... 19 P. Tirumalai , M. Lee , M. Schlansker ...

[portal.acm.org/citation.cfm?](http://portal.acm.org/citation.cfm?id=143141&jmp=indexterms&dl=portal&dl=ACM&CFID=11111111&C...)

[id=143141&jmp=indexterms&dl=portal&dl=ACM&CFID=11111111&C...](http://portal.acm.org/citation.cfm?id=143141&jmp=indexterms&dl=portal&dl=ACM&CFID=11111111&C...) - [Similar pages](#)

[PS]Software Pipelining

File Format: Adobe PostScript - [View as Text](#)

... By Rau/Schlansker/Tirumalai. ... Software Pipelining Allan, Jones, Lee, Allan. ... So, the loop kernel is composed of one instruction, which executes all the following ...

www.wisdom.weizmann.ac.il/~verify/events/2001/weekly/lecture1.ps - [Similar pages](#)

[PPT]Overview

File Format: Microsoft Powerpoint 97 - [View as HTML](#)

... Loop optimizations like software pipelining could be applied ... code generated shows no evidence of software pipelining. ... PP Tirumalai B. Ramakrishna Rau, Michael S ...

www.cse.iitd.ernet.in/esproject/homepage/docs/projects/2002-2003/diviya.final.ppt - [Similar pages](#)

[pages](#)

Software pipelining: an effective scheduling technique for VLIW ...

... references. 1 Aiken, A. and Nicolau, A. Perfect Pipelining: A New

Loop Parallelization Technique. Cornell University, Oct., 1987. ...

www.acm.org/pubs/articles/proceedings/pldi/53990/p318-lam/p318-lam.pdf - 101k - [Cached](#) -

[Similar pages](#)

[PDF]CRED: Code Size Reduction Technique and Implementation for ...

File Format: PDF/Adobe Acrobat - [View as HTML](#)

... operation corresponds to a software pipelining operation. ... conditionally executing the schedule of loop body ... we can decide the conditional registers required for ...

www.utdallas.edu/~qfzhuge/papers/escodes02-workshop.pdf - [Similar pages](#)

[PDF]Optimal Code Size Reduction for Software-Pipelined and Unfolded ...

File Format: PDF/Adobe Acrobat - [View as HTML](#)

... we need to use four conditional registers to completely ... to be inserted into the loop body for ... be parallelized with other instructions by software pipelining. ...

www.utdallas.edu/~qfzhuge/papers/issc02.pdf - [Similar pages](#)

[[More results from www.utdallas.edu](#)]

Sponsored Links

IronWorks Code Generator

Generate n-tier .NET apps with 51 customizable templates. Free trial.

www.eSolutionsPR.com

Interest: [*****](#)

.NET Code Generation Tool

Save time on any C++, C# and VB.NET project. Free trial!

www.workstate.com

Interest: [*****](#)

RAD Code Generation tool

Template based code generation

ASP.Net, VB, C# over 50 templates

www.lockwoodtech.com

Interest: [*****](#)

Code Generation in Action

Learn how to build code generators

for a range of applications domains

www.manning.com/herrington

Interest: [*****](#)

Tirupati Venkateswara VCD

Video of the famous Balaji temple

in Tirupati during Sreematsavam

www.matchless-gifts.com

Interest: [*****](#)

Code Generation Tools

Arbitrary spec or legacy languages

Many target languages, optimization

www.semanticdesigns.com

Interest: [*****](#)

Generate Source Code

Kickstart from Westfaro Corporation

simplifies source code generation.

www.westfaro.com

Interest: [*****](#)

[See your message here...](#)

[PDF] [A Register File and Scheduling Model for Application Specific ...](#)

File Format: PDF/Adobe Acrobat - [View as HTML](#)

... simple logic units can be used to activate proper **registers** at run ... schedules cyclic DFGs (Data Flow Graphs) with resource constraints using **loop pipelining**. ...


www.sigda.org/Archives/ProceedingArchives/Dac/Dac96/papers/1996/dac96/pdffiles/04_2.pdf - [Similar pages](#)

[PDF] [Controlling Code Size of Software-Pipelined Loops On the ...](#)

File Format: PDF/Adobe Acrobat - [View as HTML](#)

... support beyond the availability of static predicate **registers**. ... 3.2 Safety Considerations
For a **loop** to be eligible for software **pipelining**, the compiler ...

www.pdci.eng.wayne.edu/msp01/paper4.pdf - [Similar pages](#)

Google 

Result Page: 1 2 3 [Next](#)

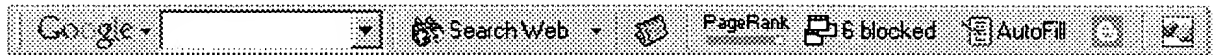
Tirumalai and pipelining and loop an

[Google Search](#)

[Search within results](#)

Dissatisfied with your search results? [Help us improve.](#)

Get the [Google Toolbar](#):



[Google Home](#) - [Advertise with Us](#) - [Business Solutions](#) - [Services & Tools](#) - [Jobs, Press, & Help](#)

©2003 Google

79 citations found. Retrieving documents...

B. R. Rau, M. Lee, P. P. Tirumalai, and M. S. Schlansker. *Register allocation for software pipelined loops*. In Proceedings of the ACM SIGPLAN '92 Conference on Programming Language Design and Implementation, pages 283–299, San Francisco, California, June 17–19, 1992. SIGPLAN Notices, 27(7), July 1992.

CiteSeer [Home/Search](#) [Document Not in Database](#) [Summary](#) [Related Articles](#) [Check](#)

This paper is cited in the following contexts:

[Documents 51 to 79](#) [Previous 50](#)

[Advanced Vector Architectures - Espasa \(1997\)](#) [\(Correct\)](#)

....needed. On top of that, program transformations such as loop blocking [PHH89, WL91, KM92, LRW91b, Li95, CM95] have proven very useful to fit the working set of a program into multilevel memory hierarchies. **Introduction 9 Related to data caching, software pipelining [Lam88, GHW90, GAG94, Jai91, RLTS92, Ram94, Rau94] has also contributed to hide memory latency and the penalties associated with cache misses by overlapping several iterations of a single loop.** Decoupling Decoupled scalar processors [SWP86, Smi84, KHC94] have focused on numerical computation and attack the memory latency problem

B. R. Rau, M. Lee, P. P. Tirumalai, and M. S. Schlansker. *Register allocation for software pipelined loops*. In Proceedings of the ACM SIGPLAN '92 Conference on Programming Language Design and Implementation, pages 283–299, San Francisco, California, June 17–19, 1992. SIGPLAN Notices, 27(7), July 1992.

[Unrolling-Based Optimizations for Modulo Scheduling - Lavery, Hwu \(1995\)](#) [\(16 citations\)](#) [\(Correct\)](#)

....more simultaneously live values exist than physical registers, spill code must be added and can significantly increase the achieved II of the loop. **In this case, it may be possible to achieve a better final II by increasing the candidate II and attempting to schedule the original loop body again [26].** If a lower bound on the loop's final register requirement for a given II were available, it would be useful during both optimization and scheduling. **During** optimization it could be used to stop optimization before excessive register pressure is generated. **During** scheduling, the candidate II's

B. R. Rau, M. Lee, P. P. Tirumalai, and M. S. Schlansker, "Register allocation for software pipelined loops," in Proceedings of the ACM SIGPLAN '92 Conference on Programming Language Design and Implementation, pp. 283–299, June 1992.

[Lifetime-Sensitive Modulo Scheduling - Huff \(1993\)](#) [\(99 citations\)](#) [\(Correct\)](#)

....the successive outputs of an operation can be kept in distinct registers. **In the absence of hardware support, the loop may be unrolled and the duplicate register specifiers renamed appropriately [9]. However, this modulo variable expansion technique can result in a large amount of code expansion [18].** A rotating register file can solve this problem without duplicating code. **Consider** saving the series of values generated by an operation in its own infinite pushdown stack. **Old** values can be read out of anywhere in the stack, and new values can be pushed on top, but a value cannot be modified

....around a vector of length II. In any case, the LiveVector's maximum, MaxLive, is the desired lower bound. **Allocating** registers for a modulo scheduled loop is beyond the scope of this paper. **For an extensive discussion of the problem, including heuristic solutions and empirical results, consult [18].** One of the most remarkable results reported in that paper is the ability of their allocation strategies to almost always achieve the MaxLive lower bound on a schedule's register pressure 4. Due to that result, this paper approximates a schedule's register pressure with its MaxLive lower bound.

B. R. Rau, M. Lee, P. P. Tirumalai, and M. S. Schlansker. *Register allocation for software pipelined loops*. In Proceedings of the ACM SIGPLAN '92 Conference on Programming Language Design and Implementation, pages 283–299, June 1992.

[Evaluation of Pseudo Vector Processor based on.. - Nakamura.. \(1994\)](#) [\(Correct\)](#)

....previous pfd just because its load pipeline has three stages. On the contrary, our architecture adopts ordinary waiting mechanism for requested data. **Due to this fact, our architecture does not need serious changes in the architecture. Modulo scheduling on rotating register files is proposed in [RLTS92].** In rotating register files, logical register number is apart from physical register number. In this point, rotating register files are similar to our slidewindowed registers. **However,** in rotating register files, the total number of physical registers is not increased. **Therefore,** long memory

B.R.Rau, M.Lee, P.P.Tirumalai, and M.S.Schlansker, "Register Allocation for Software Pipelined Loops", Proc. ACM SIGPLAN '92 Conf. on Programming Language Design and Implementation, pp.283-299, 1992

Register Allocation for Predicated Code - Eichenberger, Davidson (1995) (5 citations) (Correct)

....a framework based on cyclic interval graphs, introducing the notion of time in the register allocator paradigm. **This additional notion of time is particularly useful for the live ranges of a loop, where live ranges may cross the boundary of an iteration. Another approach, investigated by Rau et al. [12], proposes a general framework for the allocation of registers in software pipelined loops for various code generation and hardware support schemes.** The second contribution of this paper is a set of heuristics that reduces the register requirements by allowing non interfering virtual registers

....For register allocators based on Chaitin s graph coloring framework [9] 10] register allocation for predicated code can be achieved simply by using the refined interference graph instead of the conventional one. **However, several register allocators depart from the graph coloring method [11][12] as graph coloring methods do not provide a notion of time that is particularly useful for the live ranges of a loop, which may cross the boundary of an iteration.** Also, nontraditional constraints such as the one presented in [12] to support various code generation and hardware support schemes are

[Article contains additional citation context not shown here]

B. R. Rau, M. Lee, P. P. Tirumalai, and M. S. Schlansker. *Register allocation for software pipelined loops*. PLDI, pages 283--299, June 1992.

An Integrated Approach to Register Binding and Scheduling - Bart Mesman (Correct)

....to satisfy the timing constraints, software pipelining [2] also called loop pipelining or loop folding, is required. **Previously [15] we showed that a heuristic like list scheduling for loop pipelining is unable to satisfy the timing and resource constraints even for simple examples. Rau et al. [11] successfully perform register binding tuned to pipelined loops.** They mention that for better code quality Concurrent scheduling and register allocation is preferable , but for reasons of run time efficiency they solve the problem of scheduling and register binding in separate phases. **Some**

B.R. Rau, M. Lee, P.P. Tirumalai and M.S. Schlansker, "Register allocation for software pipelined loops", Proc. of the SIGPLAN '92 conf. on Programming language design and implementation, pp. 283-299, June 1992

Minimizing Register Requirements under... - Govindarajan, Altman.. (1994) (40 citations) (Correct)

....Huff s Slack Scheduling [9] Wang, Eisenbeis, Jourdan and Su s FRLC [23] and Gasperoni and Schwiegelshohn s modified list scheduling [6] Experimental results show that the method described in this paper performed significantly better than these methods. **1 Introduction Software pipelining [1, 4, 9, 11, 12, 13, 17, 18, 22] has been proposed as an efficient method for loop schedul This work was supported by research grants from NSERC (Canada) and MICRONET Network Centers of Excellence (Canada) To Appear in the Proceedings of the 27th Annual International Symposium on Microarchitectures (MICRO 27) San Jose,**

....in Section 7. 2 Exploiting the Space of Software Pipelined Schedules 2. **1 An Example We introduce the notion of rate optimal schedules under resource constraints, and illustrate how to search among them the ones which optimize the register usage with the help of a simple example loop taken from [18].** The loop L (in the C language) is: for (i = 0; i < n; i) f s = s a[i] a[i] s s a[i] g The dependence graph for the loop L is depicted in Figure 1. S0 S1 S2 S3 S4 S5 Figure 1: Dependence Graph of Loop L Consider an architecture with 3 pipelined homogeneous function units. **Assume**

[Article contains additional citation context not shown here]

B. R. Rau, M. Lee, P. P. Tirumalai, and M. S. Schlansker. *Register allocation for software pipelined loops*. In Proc. of the SIGPLAN '92 Conf. on Programming Language Design and Implementation, pages 283--299, San Francisco, CA, Jun. 17--19, 1992.

Towards Identifying and Monitoring Optimization Impacts - Way, Pollock (1997) (2 citations) (Correct)

....is rarely gathered and exploited in the optimizer s strategy. **There are isolated instances where this information is used to good effect, such as when combining instruction scheduling and register allocation [3, 5, 6, 19, 20, 30, 33, 34] or software pipelining and register allocation [16, 17, 21, 23, 27, 32, 38, 44].** While these techniques can improve program performance, they focus narrowly on the interaction of a single pair of optimizations, rather than more generally on the entire collection of optimizations to be applied to a program. **Provided that enough useful information can be gathered and analyzed**

....of balance among the levels of demand for specific machine resources of particular interest to the two phases, and the supply and configuration of the target machine s resources. **The most well known examples of this work focus on the interactions between software pipelining register allocation [16, 17, 21, 23, 27, 32, 38, 44], instruction scheduling and register**

allocation [3, 5, 6, 19, 20, 30, 33, 34] instruction scheduling and cache usage [28] and scalar replacement and register allocation [8] All have in common the goal of creating a good match between the program characteristics, such as instruction placement

B. R. Rau, M. Lee, P. P. Tirumalai, and M. S. Schlansker. *Register allocation for software pipelined loops*. In ACM SIGPLAN Conference on Programming Language Design and Implementation, 1992.

Non-Consistent Dual Register Files to Reduce Register Pressure - Llosa, Valero, Ayguade (1995) (8 citations) (Correct)

....Chaitin's technique based on graph coloring[14] Register allocation for software pipelined loops presents additional problems leading to unconventional solutions. **How to allocate registers for modulo scheduled loops is beyond the scope of this paper (for an extensive discussion of the problem see [15]) The Wands Only strategy combined with the First Fit allocation schema have been chosen to allocate registers.** Wands Only is the strategy that has the lowest empirical complexity, and the one that obtains the more optimal results in terms of number of registers. For this strategy all the

....M3 and A4. The results of M3 are used by operation A4; since A4 has been scheduled in 5 We have chosen this example because it is very simple to calculate the registers required by the schedule. **For an extensive discussion of the register allocation problem for software pipelined loops see [15].** VALUE L1 L2 M3 A4 M5 A6 Allocation GL LO LO RO RO RO Lifetime 13 7 6 6 6 4 Table 3: Allocation requirements of values for example loop. the left cluster, the values produced by M3 could be allocated as left only values. The results of A4 are used by operation M5; since M5 has been scheduled

B.R. Rau, M. Lee, P. Tirumalai, and P. Schlansker. *Register allocation for software pipelined loops*. In Proceedings of the ACM SIGPLAN '92 Conference on Programming Language Design and Implementation, pages 283-299, June 1992.

Constraint Driven Approach To Loop Pipelining And... - Mesman, Strik... (1998) (Correct)

....to satisfy the timing constraints, software pipelining [2] also called loop pipelining or loop folding, is required. **Previously [15] we showed that a heuristic like list scheduling for loop pipelining is unable to satisfy the timing and resource constraints even for simple examples. Rau et al. [11] successfully perform register binding tuned to pipelined loops.** They mention that for better code quality Concurrent scheduling and register allocation is preferable, but for reasons of run time efficiency they solve the problem of scheduling and register binding in separate phases. **Some**

B.R. Rau, M. Lee, P.P. Tirumalai and M.S. Schlansker, "Register allocation for software pipelined loops", Proc. of the SIGPLAN '92 conf. on Programming language design and implementation, pp. 283-299, June 1992

A Scalar Architecture for Pseudo Vector Processing based on... - Nakamura Hiroshi (1 citation) (Correct)

....three stages. **Compared with i860 architecture, our architecture includes ordinary waiting mechanism for requested data and successfully closes the growing gap between processor and memory speed without serious changes in the architecture. Modulo scheduling on rotating register files is proposed in [RLTS92].** In rotating register files, logical register number is apart from physical register number. This is similar to our slide windowed registers. **However, in rotating register files, the total number of physical registers is not increased. Therefore, long memory access latency cannot be hidden. This**

B.R. Rau, M. Lee, P.P. Tirumalai, and M.S. Schlansker, "Register Allocation for Software Pipelined Loops", Proc. ACM SIGPLAN '92 Conf. on Programming Language Design and Implementation, pp.283-299, 1992

Array Data Flow Analysis for Load-Store Optimizations in... - Bodík, Gupta (1995) (2 citations) (Correct)

No context found.

B. R. Rau, M. Lee, P. P. Tirumalai, M. S. Schlansker, "Register Allocation for Software Pipelined Loops," Proc. of the SIGPLAN Conference on Programming Language Design and Implementation, San Francisco, California, pages 212-223, June 1992.

[Documents 51 to 79](#) [Previous 50](#)

[Online articles have much greater impact](#) [More about CiteSeer](#) [Add search form to your site](#) [Submit documents](#) [Feedback](#)

CiteSeer - citeseer.org - [Terms of Service](#) - [Privacy Policy](#) - Copyright © 1997-2002 NEC Research Institute